Ambika Koushik

M12507845

CS 5158/6058 Data Security and Privacy, Spring 2018

Project 1: One-Time Pad

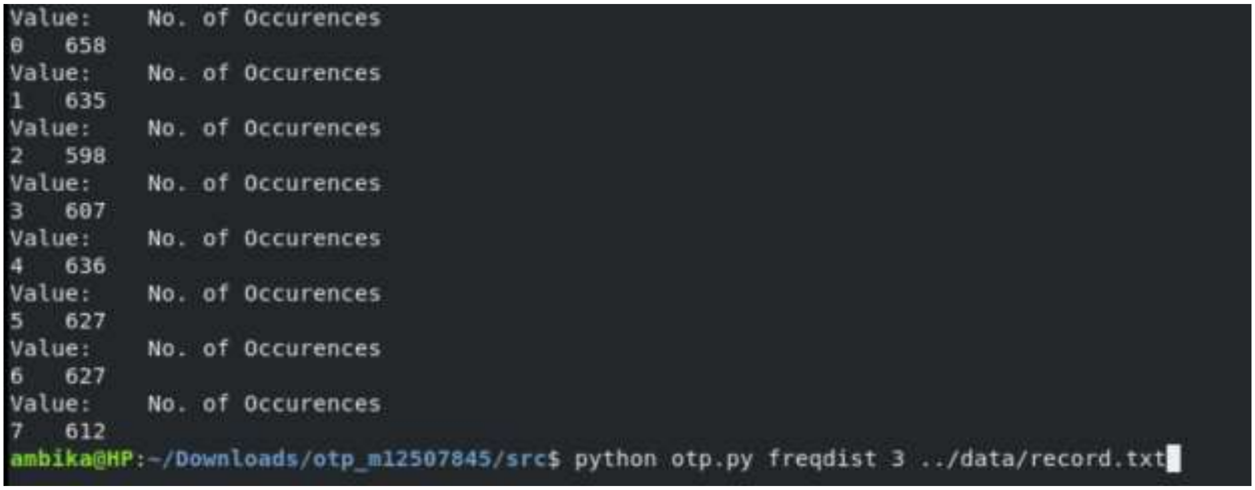
4. (CS 6058 Only) Distribution of Keys:

(a) Given a security parameter λ = 3, repeat your key generation function for at least 5000 times. Record all the unique 3-bit keys you program generated, calculate the frequency of each one, and prove that those keys are (almost) uniformly distributed. You should have a function to automatically collect this frequency distribution. In addition, you also need to evaluate the average running time of your encryption function with λ = 128.

(b) You need to submit one-page report (in pdf) to explain and show your results in terms of key distribution and encryption time. In your report, you should describe details of your implementation, such as OS, programming language, crypto libraries, encryption parameters, etc. You can use tables, figures or screenshots to help you present your results in your report. Please put this report under your project folder and submit it together with your code.

Solution: I am using **os**, **sys** and **random** libraries.

We have written a function namely "keygen" which will not only generate the 3-bit keys but also gets the frequency of them. From the below screenshot you can observe that the 3bits keys are uniformly distributed.

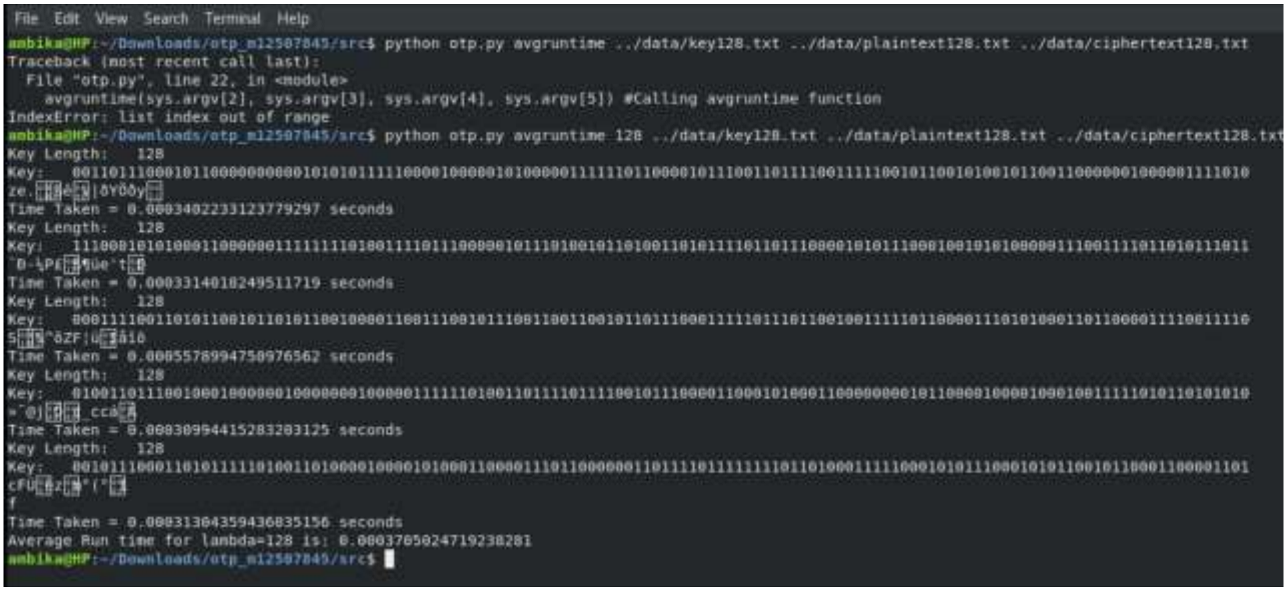


```
Value:    No. of Occurences
0   658
Value:    No. of Occurences
1   635
Value:    No. of Occurences
2   598
Value:    No. of Occurences
3   607
Value:    No. of Occurences
4   636
Value:    No. of Occurences
5   627
Value:    No. of Occurences
6   627
Value:    No. of Occurences
7   612
ambika@HP:~/Downloads/otp_m12507845/src$ python otp.py freqdist 3 ../data/record.txt
```

For the second case where, the key length is 128bits, I have taken a **plaintext128.txt** file which has the content “Ms.AmbikaKoushik” which is of length 128bits. The Keygen function will create a 12bit key and that will be stored in the key128.txt file. After encryption, the respective ciphertext will be displayed on the command prompt after execution and will be stored in **ciphertext128.txt** file. Also, for the 128bit key I have ran the encryption function for 5times and calculated the average running time. When we run the “avgruntime” function, it displays the key value (which is 128bits), its corresponding encrypted text and the time taken for that run. At the end we can see the average running time taken for the encryption function. (A screenshot is also provided for better understanding)

The output is as below:

```
•   Key Length:   128
    Key:   00110111000101100000000000101010111110000100000101000001111110110000101110011
    01111001111100101100101001011001100000001000001111010
    ze. àê v|ðYÖðy
•   Time Taken = 0.0003402233123779297 seconds
    Key Length:   128
    Key:   1110001010100011000000111111110100111101110000010111010010110100110101111011
    0111000010101110001001010100000111001111011010111011
    ¯Ð-¼P£ ß¶üe'tŸÐ
    Time Taken = 0.0003314018249511719 seconds
•   Key Length:   128
    Key:   0001111001101011001011010110010000110011100101110011001100101101110001111101
    1101100100111110110000111010100011011000011110011110
    S %^õZF¦ü™Iåîõ
    Time Taken = 0.0005578994750976562 seconds
•   Key Length:   128
    Key:   01001101110010001000000010000000100000111111010011011110111100101110000110001
    0100011000000001011000010000100010011111010110101010
    »¯@j‹ÔŽ¢_ccáœÁ
    Time Taken = 0.00030994415283203125 seconds
•   Key Length:   128
    Key:   0010111000110101111101001101000010000101000110000111011000000110111101111111
    1011010001111100010101110001010110010110001100001101
    cFÚ‘èz m°(° 1
    f
    Time Taken = 0.00031304359436035156 seconds
    Average Run time for lambda=128 is: 0.0003705024719238281
```



```
ambika@HP:~/Downloads/otp_m12507845/src$ python otp.py avgruntime ../data/key128.txt ../data/plaintext128.txt ../data/ciphertext128.txt
Traceback (most recent call last):
  File "otp.py", line 22, in <module>
    avgruntime(sys.argv[2], sys.argv[3], sys.argv[4], sys.argv[5]) #Calling avgruntime function
IndexError: list index out of range
ambika@HP:~/Downloads/otp_m12507845/src$ python otp.py avgruntime 128 ../data/key128.txt ../data/plaintext128.txt ../data/ciphertext128.txt
Key Length:   128
Key:   00110111000101100000000000101010111110000100000101000001111110110000101110011011110011111001011001010010110011000000010000001111010
ze.□□é□|ðYõöy□
Time Taken = 0.0003402233123779297 seconds
Key Length:   128
Key:   11100010101000110000001111111101001111011100000101110100101101001101011110110111000010101110001001010100000111001111011010111011
`Ð-¼PÉ□¶üe`t□
Time Taken = 0.0003314018249511719 seconds
Key Length:   128
Key:   00011110011010110010110101100100001100111001011100110011001011011100011111011101100100111110110000111010100011011000011110011110
S□□^õZF¦ü□åî6
Time Taken = 0.0005578994750976562 seconds
Key Length:   128
Key:   01001101110010001000000010000000100000111111010011011110111100101110000110001010001100000000101100001000010001001111101011010101010
=¯@j□□_ccà□
Time Taken = 0.0003099441528320312 5 seconds
Key Length:   128
Key:   00101110001101011111010011010000100001010001100001110110000000110111101111111101101000111110001010111000101011001011000110000110 1
cFU□öz□°(°□
f
Time Taken = 0.0003130435943603515 6 seconds
Average Run time for lambda=128 is: 0.0003705024719238281
ambika@HP:~/Downloads/otp_m12507845/src$ █
```